REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From - To)*
15-12-2011	Briefing Charts	

4. TITLE AND SUBTITLE

Development of a Flow Solver with Complex Kinetics on the Graphic Processing Units (GPU)

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

Hai P. Le and Jean-Luc Cambier

5d. PROJECT NUMBER

5f. WORK UNIT NUMBER
23041057

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Air Force Research Laboratory (AFMC)
AFRL/RZSS
1 Ara Drive
Edwards AFB CA 93524-7013

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Air Force Research Laboratory (AFMC)
AFRL/RZS
5 Pollux Drive
Edwards AFB CA 93524-7048

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S NUMBER(S)
AFRL-RZ-ED-VG-2011-581

12. DISTRIBUTION / AVAILABILITY STATEMENT

Distribution A: Approved for public release; distribution unlimited. PA# 111067.

13. SUPPLEMENTARY NOTES

For presentation at the 50th AIAA Aerospace Sciences Meeting, Nashville, TN, 9-12 Jan 2012.

14. ABSTRACT

In the current work, we have implemented a numerical solver on the Graphic Processing Units (GPU) to solve the reactive Euler equations with detailed chemical kinetics. The solver incorporates high-order finite volume methods for solving the fluid dynamical equations and an implicit point solver for the chemical kinetics. Generally, the computing time is dominated by the time spent on solving the kinetics which can be benefitted from the computing power of the GPSs. Preliminary investigation shows that the performance of the kinetics solver strongly depends on the mechanism used in the simulations. The speed-up factor obtained in the simulation of an ideal gas ranges from 30 to 55 compared to the CPU. For a 9-species gas mixture, we obtained a speed-up factor of 7.5 to 9.5 compared to the CPU. For such a small mechanism, the achieved speed-up factor is quite promising. This factor is expected to go much higher when the size of the mechanism is increased. The numerical formulation for solving the reactive Euler equations is briefly discussed in this paper along with the GPU implementation strategy. We also discussed some preliminary performance results obtained with the current solver.

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
					Dr. Jean-Luc Cambier
a. REPORT	b. ABSTRACT	c. THIS PAGE	SAR	30	19b. TELEPHONE NUMBER *(include area code)*
Unclassified	**Unclassified**	**Unclassified**			N/A

Development of a Flow Solver with Complex Kinetics on the Graphic Processing Units (GPU)

Hai P. Le[1], *Jean-Luc Cambier*[2]

[1]Mechanical and Aerospace Engineering Department
University of California, Los Angeles

[2]Air Force Research Laboratory
Edwards AFB, CA

January 11, 2012

50th AIAA Aerospace Sciences Meeting, Nashville, Tennessee

Table of Contents

Objectives

- Develop a fluid code on the GPU for modeling flows with complex chemical kinetics. The entire code is written using CUDA C/C++ for maximum flexibility.
- Explore different strategies for optimizing the performance of the code for a general chemistry mechanism.
  - Emphasis on the kinetics solver since it is more computationally expensive.
- Benchmark with standard test cases.

Motivation

- Detail study of non-equilibrium processes associated with high-speed flow.
  - Detonation instability
  - Partially ionized gas
  - MHD
- Development of a multi-physics code utilizing Object-Oriented and CUDA technology. Both of these features are available in CUDA C/C++.

Governing Equations

Euler equations with source term for chemical kinetics

$$\frac{\partial \mathbf{Q}}{\partial t} + \frac{1}{V}\oint_S \mathbf{F_n} dS = \dot{\mathbf{\Omega}} \tag{1}$$

$$\mathbf{Q} = \begin{pmatrix} \rho_s \\ \rho u \\ \rho v \\ \rho w \\ E \end{pmatrix} ; \mathbf{F_n} = \begin{pmatrix} \rho_s U_n \\ P n_x + \rho u U_n \\ P n_y + \rho v U_n \\ P n_z + \rho w U_n \\ U_n H \end{pmatrix} ; \dot{\mathbf{\Omega}} = \begin{pmatrix} \dot{\omega}_s \\ 0 \\ 0 \\ 0 \\ -\sum_s \dot{\omega}_s e_{0s} \end{pmatrix}$$

Solution method:

- Finite Volume method for hyperbolic conservation laws
- Source terms are solved by using operator splitting technique

Numerical Schemes

- Monoticity Preserving[1] (MP) Schemes
  - 3rd and 5th order spatial discretization was used in conjunction with 3rd order TVD-Runge-Kutta time integration
- Arbitrary Derivative Riemann solver with Weighted Essential Non-Oscillatory[2] (ADERWENO) scheme
  - 5th order spatial and 3rd order temporal without Runge-Kutta time integration
  - Utilizes Cauchy-Kowalewski procedure and Taylor series expansion of WENO fluxes for high order in time

[1] Suresh & Huynh (1997) *J. Comp. Phys.* 136, 83-99
[2] Titarev & Toro (2001) *J. Comp. Phys.* 204, 715-736

Chemical Kinetics

Implicit formulation

$$\frac{d\mathbf{Q}}{dt} = \dot{\mathbf{\Omega}} \rightarrow \left(I - \Delta t \frac{\partial \dot{\mathbf{\Omega}}}{\partial \mathbf{Q}} \right) \frac{d\mathbf{Q}}{dt} = \dot{\mathbf{\Omega}} \tag{2}$$

Elementary Reaction:

$$\sum_s \nu'_{rs}[X_s] \rightleftarrows \sum_s \nu''_{rs}[X_s] \tag{3}$$

Species production/destruction rate

$$\dot{\omega}_s = \sum_r \nu_{rs} K_{fr} \prod_s [X_s]^{\nu'_{rs}} - \sum_r \nu_{rs} K_{br} \prod_s [X_s]^{\nu''_{rs}} \tag{4}$$

where

$$\nu_{rs} = \nu''_{rs} - \nu'_{rs}$$

Graphic Processing Unit

What is GPU?

- Graphic processing units containing a massive amount of processing cores
- Designed specifically for graphic rendering which is a highly parallel process

Why GPU?

- GPU is faster than CPU on SIMD execution model
- GPU is now very easy to program
- GPU is much cheaper than CPU

GPU versus CPU

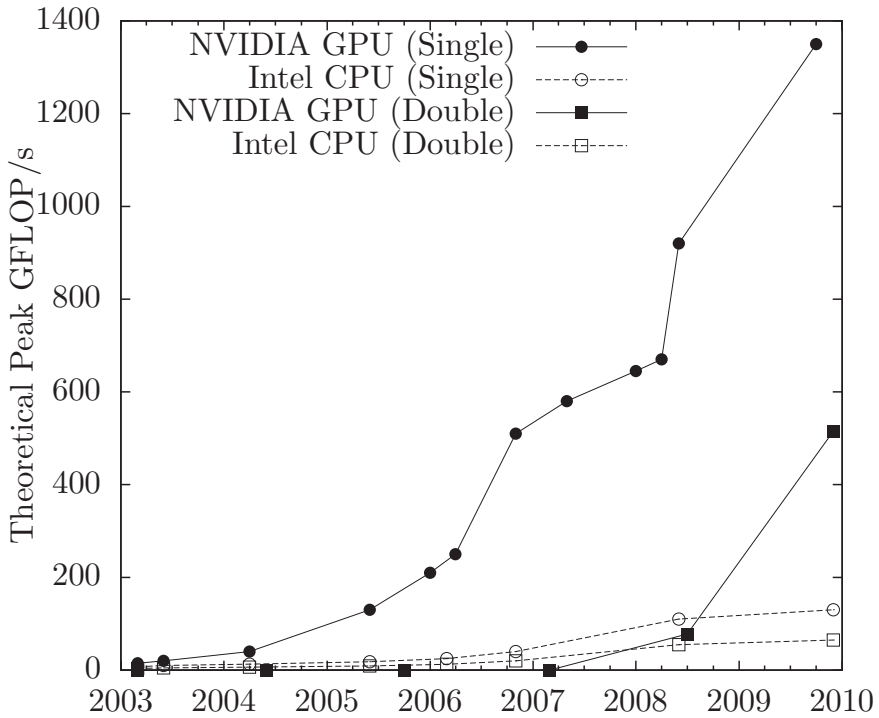


Figure: Single and double precision floating point operation capability of GPU and CPU from 2003-2010

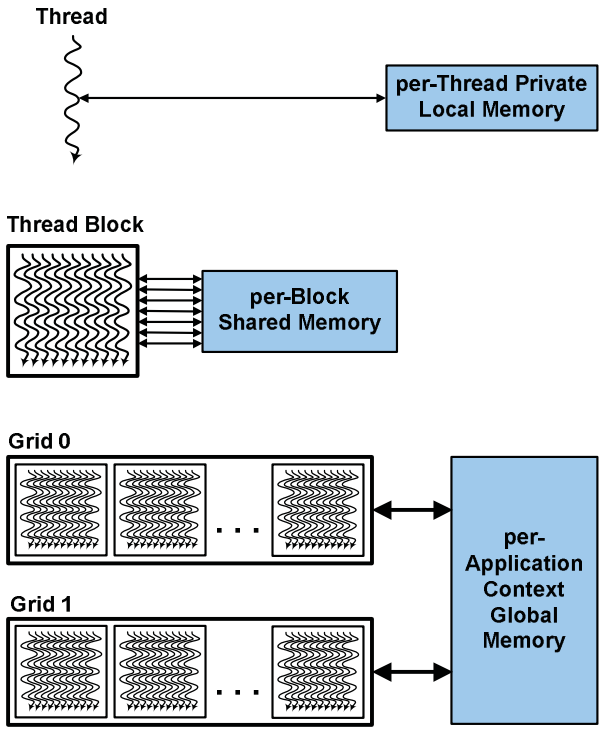
GPU Programming

Programming languages for GPU: CUDA, OpenCL, DirectCompute, BrookGPU, ...
CUDA is the most mature programing environment for GPU.

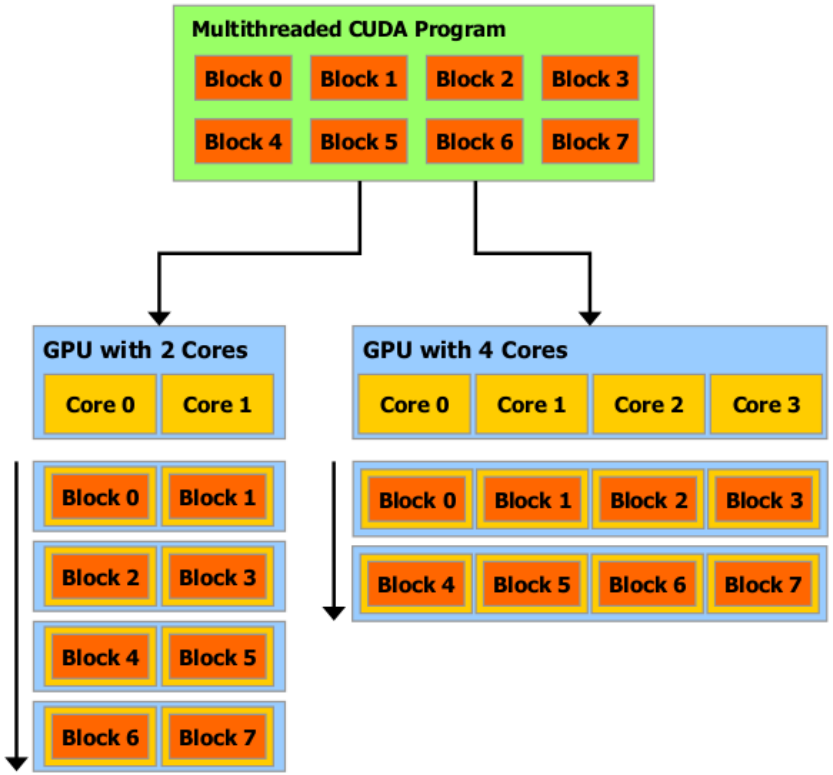
- similar to C/C++
- support OO features
- easy to debug

GPU Programming Model

- Each device contains a set of streaming multi-processor (SM). Each SM contains a set of streaming processors (SP).
- Parallel based on *grid* and *thread blocks*
- Execution instruction called *kernel*



GPU Programming Model



CFD

CFD:

- Cell-based parallelization: EOS, time marching, etc.
- Face-based parallelization: Reconstruction, flux, etc.

Strategies:

- Global memory
  - large but high latency; requires coalesced access
- Shared memory
  - small but very fast; not useful in this case since $N_Q \sim N_s$
- Reduce block occupancy to utilize more registers[3].

[3] Volkov (2010) Better Performance at Lower Occupancy, *GPU Tech. Conf.*

Chemical Kinetics

Main strategies

- Coalesce memory access pattern for high global memory bandwidth
- Utilize shared memory to reduce DRAM latency
- Texture binding for read-only data

Issues:

- How to overcome shared memory limitation?
- How effective is global memory?

Summary of Steps in Gaussian Elimination Algorithm

- Forward substitution:

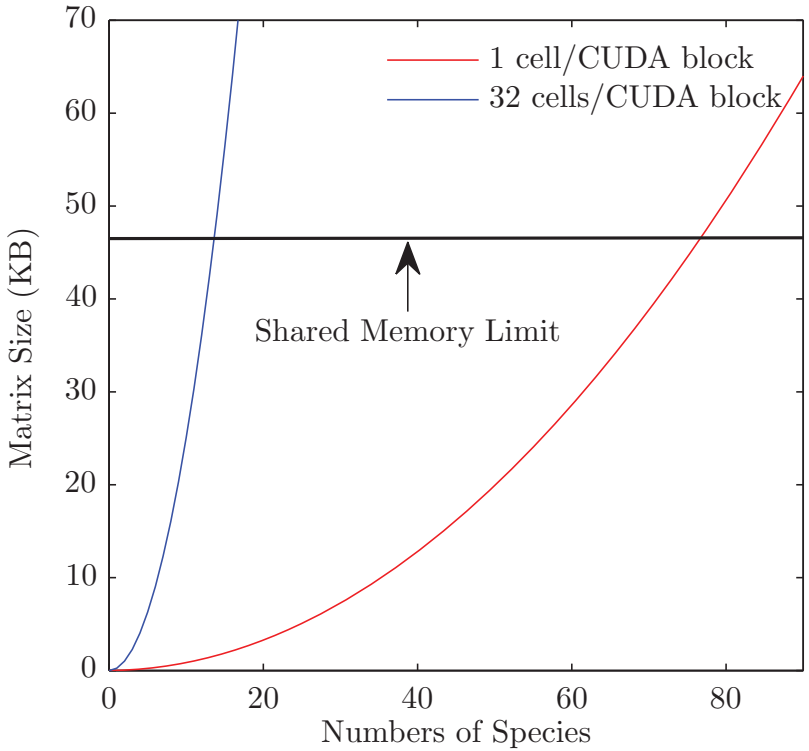
```
for np = 1:N-1
  for ns = np+1:N
      P := A(ns,np)/A(np,np)
      RHS(ns) := RHS(ns)-RHS(np)*P
      for ms = np+1:N
          A(ns,ms) := A(ns,ms)-A(np,ms)*P
```

- Backward substitution:

```
for np = N-1:1
  P := 0
  for ns=np+1:N
      P := P+A(np,ns)*RHS(ns)
  RHS(np) := (RHS(np)-P)/A(np,np)
```

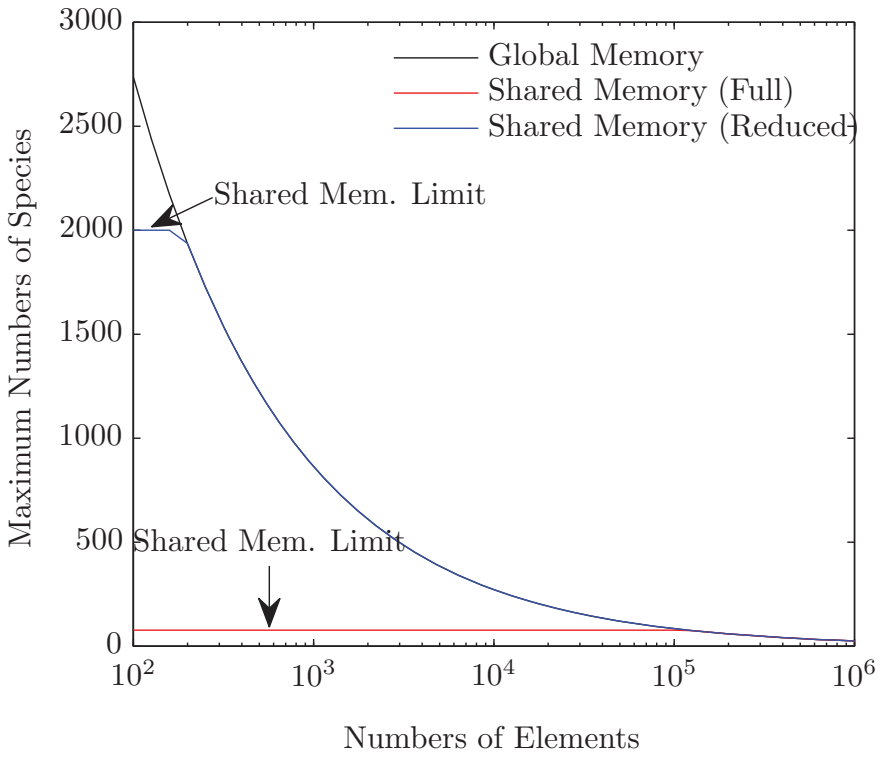
Shared Memory Limit

How many kinetics system can we put on shared memory (48 KB/CUDA block)?



Reduced Storage Pattern

Store one row of matrix in shared memory for each row elimination

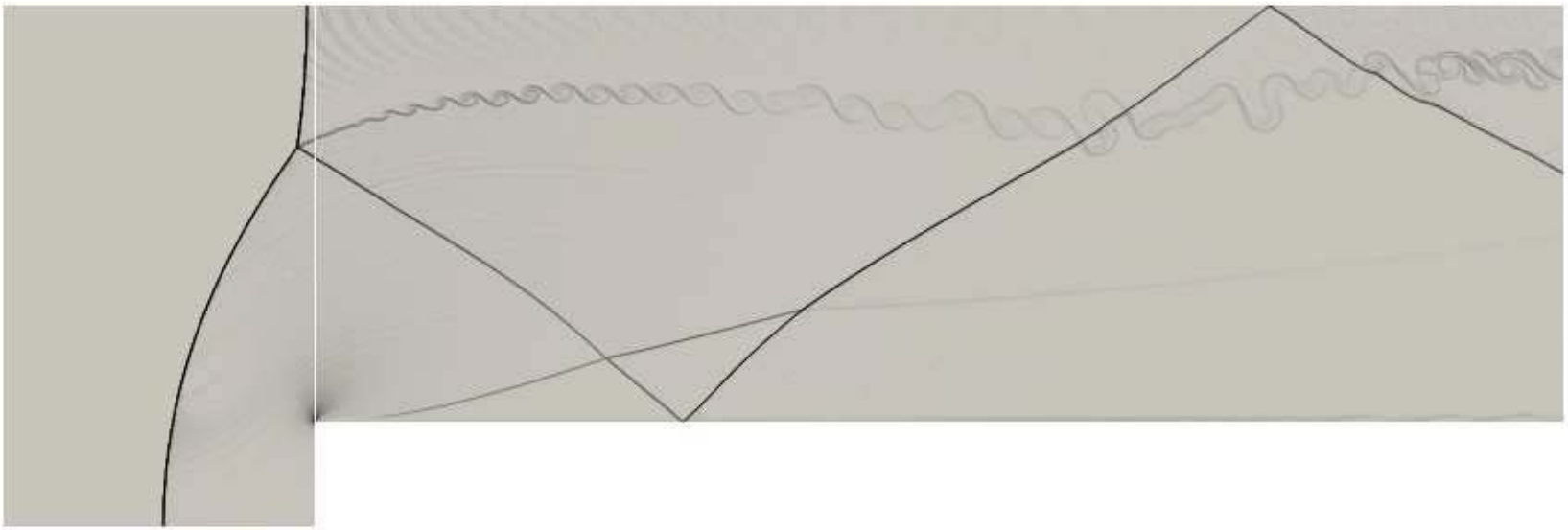


Algorithms

- Algorithm 1: store matrix data on global memory and coalesce memory access pattern
  - Inverse several matrices per CUDA block
- Algorithm 2: store part of matrix data (one row at a time) on shared memory
  - Load and reload after row pivoting
  - Inverse one matrix per CUDA block

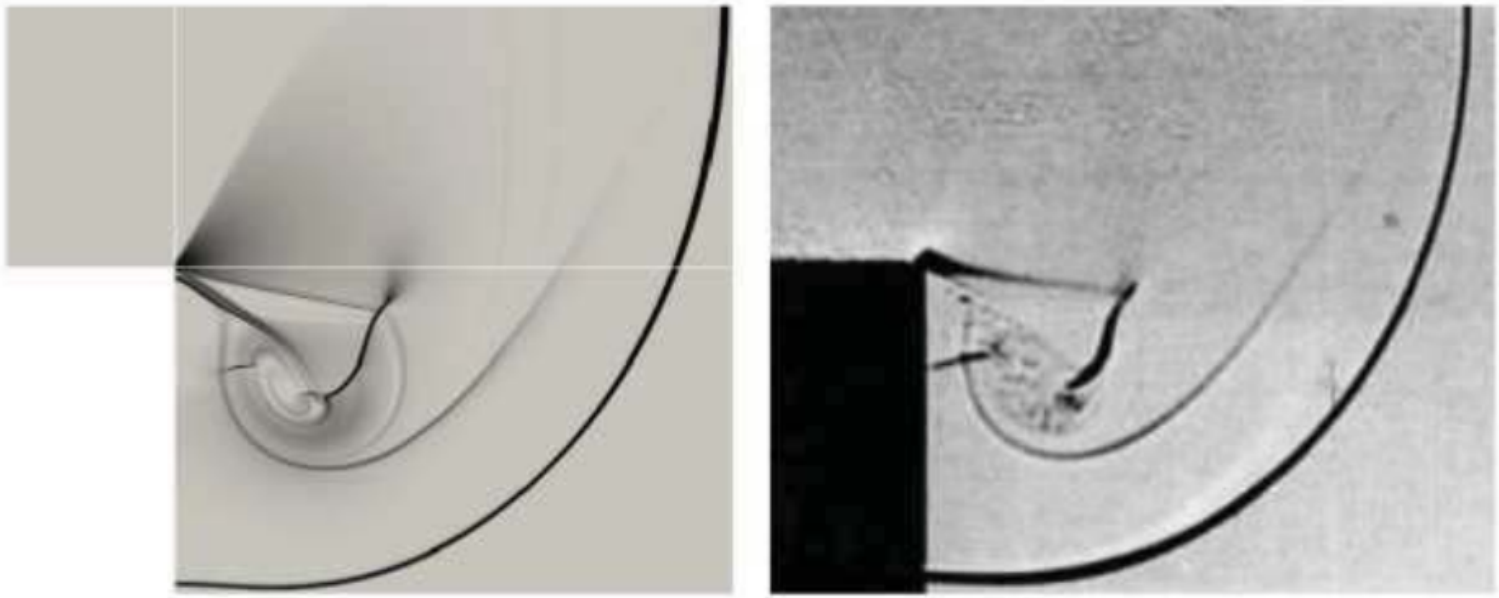
CFD Results: Forward Step

- Mach 3 flow over a step with reflective boundary on top
- No special treatment at the corner of the step
- MP5 scheme with RK3 using 630,000 cells



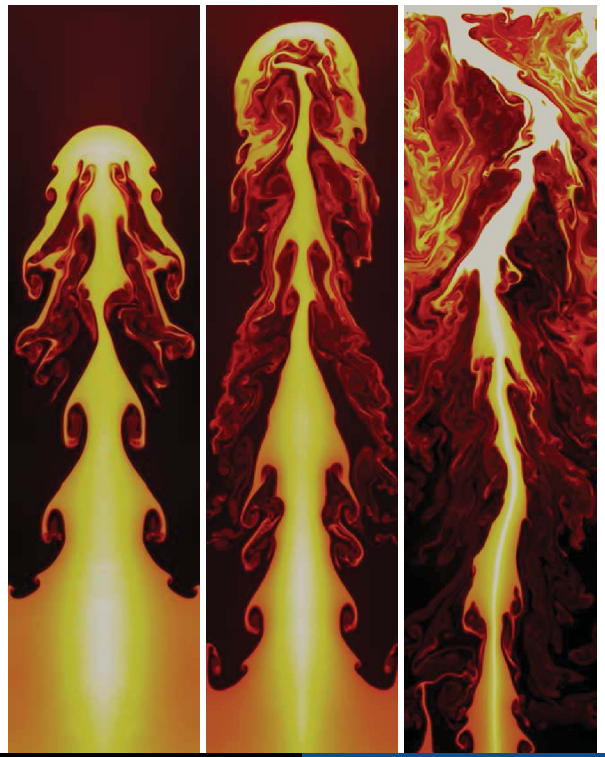
CFD Results: Backward Step

- Mach 2.4 shock diffracted from a step
- MP5 scheme with RK3 using 300,000 cells
- Comparison with experiment shows excellent agreement



CFD Results: Rayleigh-Taylor Instability

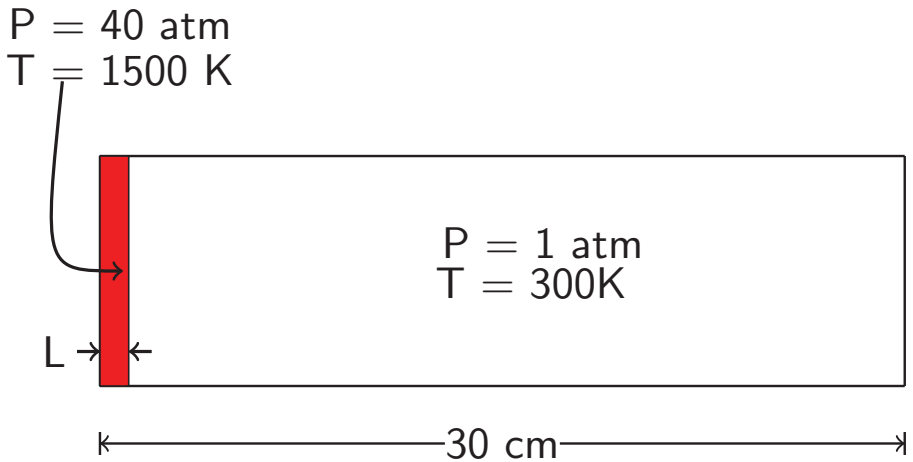
- Acceleration of a heavy fluid to a lighter fluid
- MP5 scheme with RK3 using 1.6M cells
- Contact discontinuity well resolved; evidence of fine scale instability structure



Cellular Detonation

Test setup:

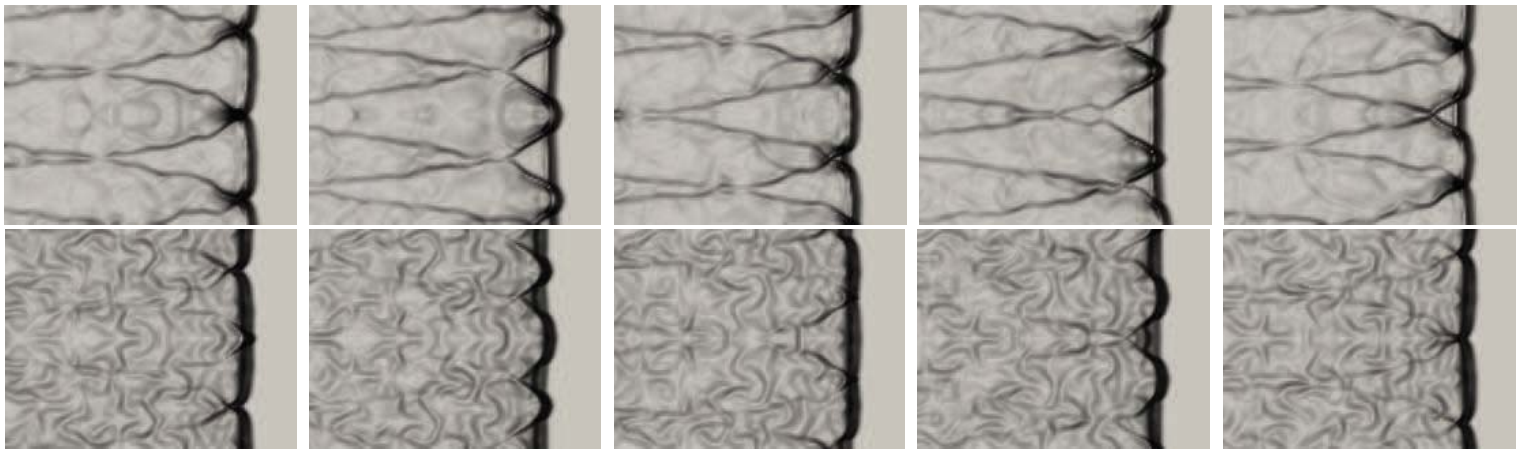
- Wall sparked ignition ($P = 40$ atm; $T = 1500$ K) with premixed Stoichiometric Mixture of H_2Air
- Contact discontinuity initially disturbed in 2-D simulation
- Maas and Warnatz[4] H_2-O_2 reaction mechanism



[4] Maas, U. and J. Warnatz (1988). *Combust. Flame* 74, 5369.

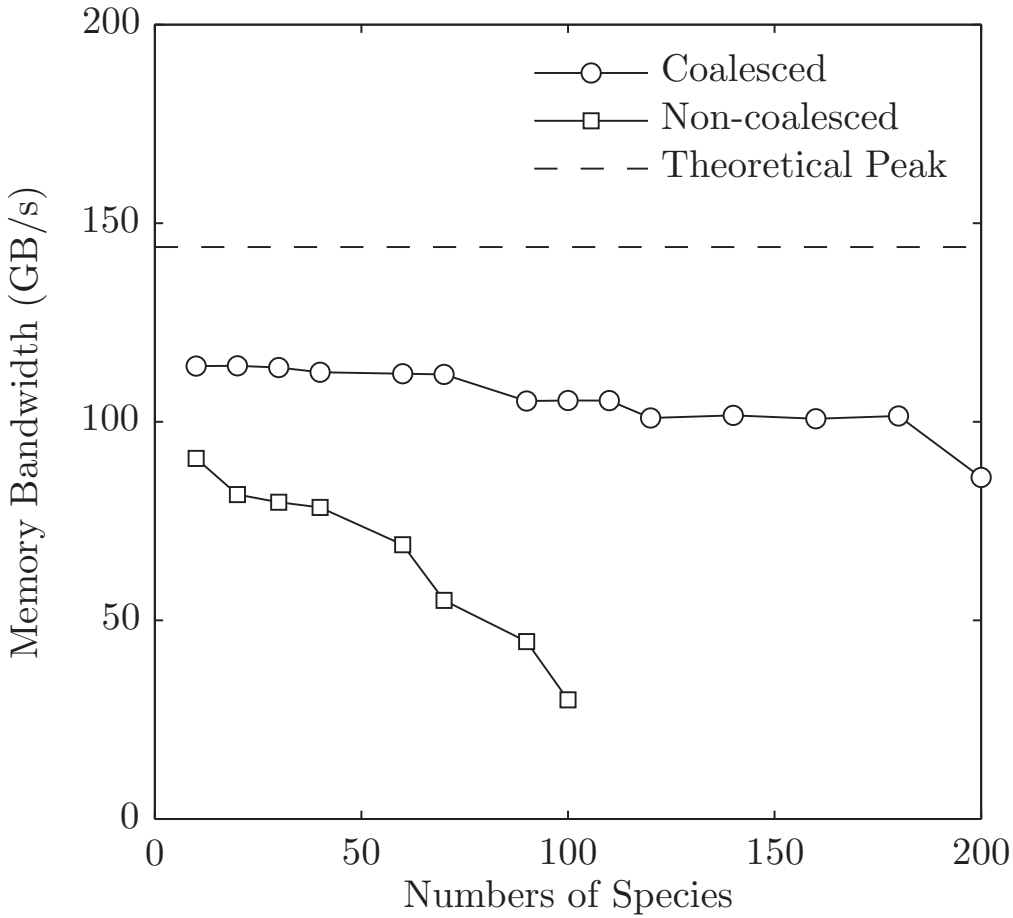
Cellular Detonation

- Pressure and temperature evolution of flow field
- Cellular structure developed due to to flame front instability



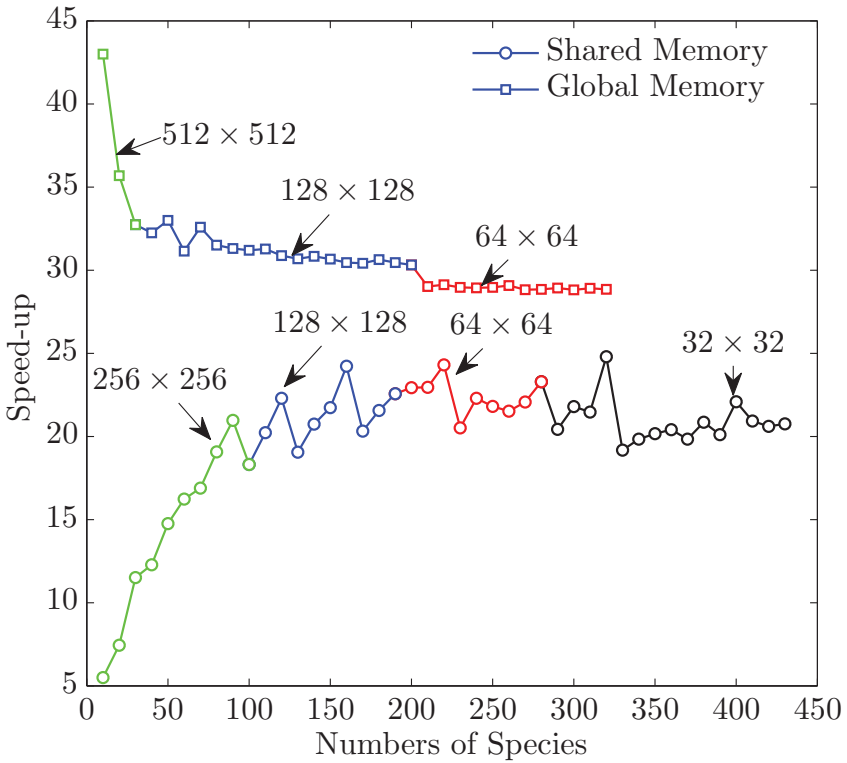
Performance Results: Algorithm 1

How effective is global memory access?



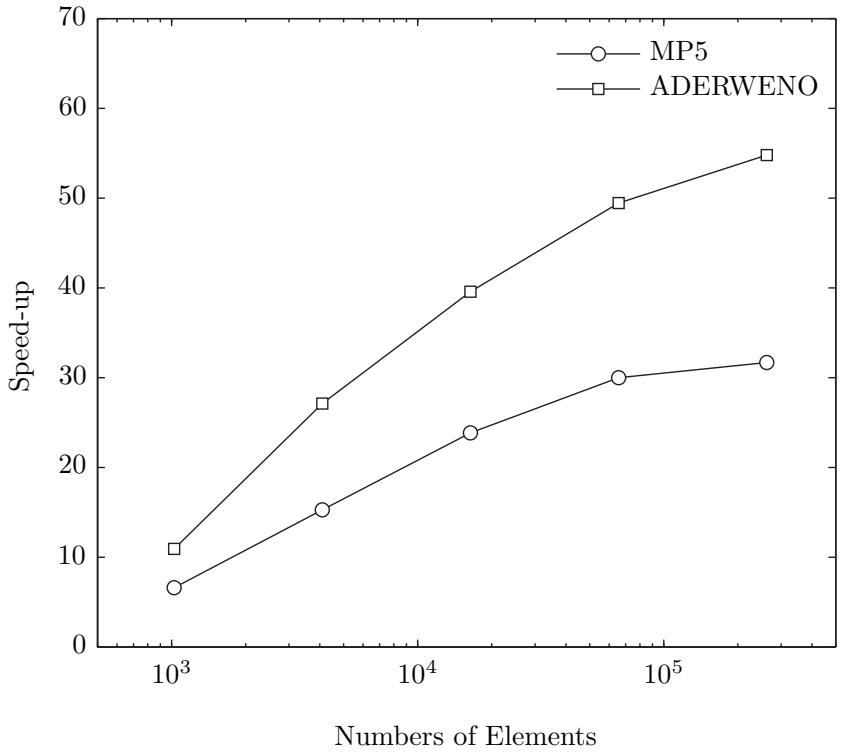
Performance Results: Algorithm 1 vs. 2

- Measurement of the performance of the kinetics solver for different species sizes.
- Grid size is varied due to limitation of global memory



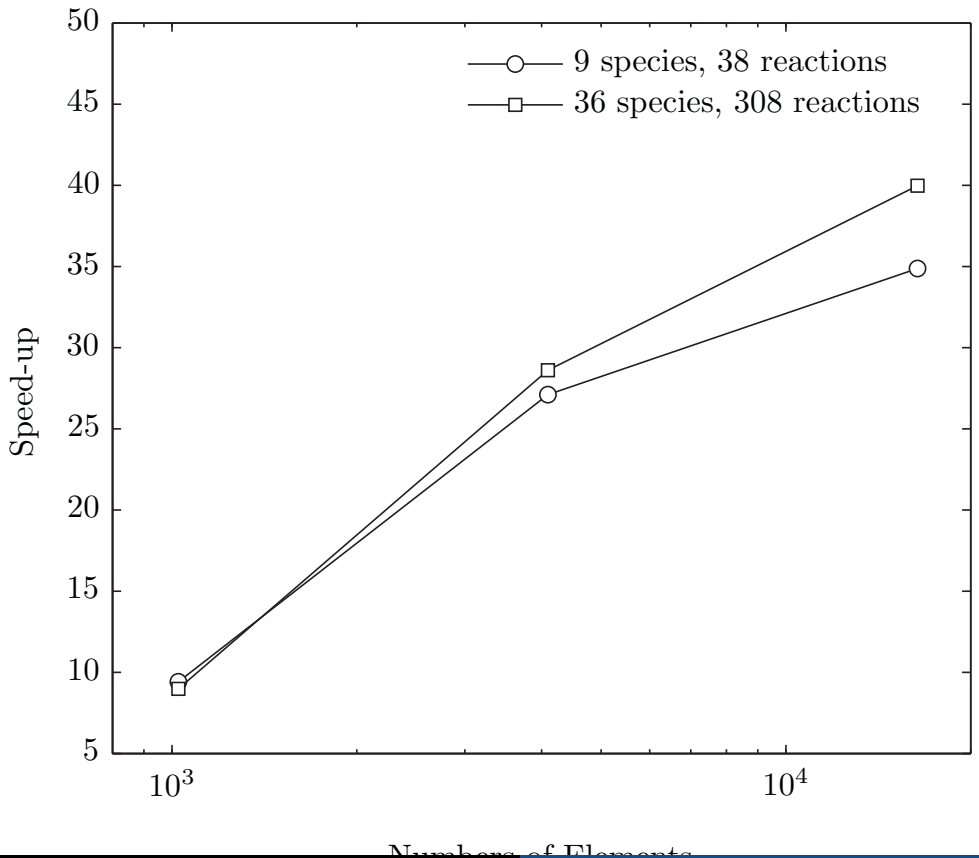
Performance Results: CFD

- ADERWENO shows substantial advantages over the MP5 due to single step integration



Performance

- Speed-up obtained for a larger mechanism ($CH_4 - O_2$) is nearly 40 times faster



Conclusion and Future Works

Accomplishment:

- Basic CFD framework for fluid simulation with detailed chemical kinetics.
- Performance obtained in both cases are very promising: up to 60 times for non-reacting flow and up to 40 for reacting flow

Future Works:

- Extension to Multi-GPU using MPI
- Collisional-Radiative kinetics for partial ionized gas
- MHD simulation for electromagnetic field effects

Acknowledgements and Questions

AFRL

- Mr. David Bilyeu
- Dr. Justin Koo

UCLA

- Mr. Lord Cole
- Prof. Ann Karagozian

Questions?

- Hai Le
  hai.le@ucla.edu